Rendezések alapprogram

A továbbiakban különböző rendezéseket készítenénk és vizsgálnánk meg. A rendezések úgynevezett helyben rendezések, ami azt jelenti, hogy néhány egyéb változótól eltekintve nem hoznánk létre még egyszer a sorozat elemeit, hanem az eredeti adatsorozatban cserélgetjük, tologatjuk őket.

Készítsünk a repo-ban egy új Projektet Studio Code segítségével (**Ctrl-Shift-P, Console**) a neve legyen **Rendezések**! Az ebben lévő Program.cs állományba dolgozzunk mindegyik rendezés során!

Hozzunk létre egy alapprogramot a rendezésekhez! Legyen a szokásos `using System;` után egy osztály, például `class Rendezések {}`. Ebben az osztályban hozzunk létre egy tömböt, amit majd rendezünk, illetve egy konstans értéket, ami mutatja, hogy hány elemű a tömb!

```
    static int [] tomb;

    const int N = 30;
```

A tömbnek majd a főprogramban foglalunk helyet a `new` paranccsal, most csak megadtuk, hogy tetszőleges metódus az osztályon belül el tudja érni. A főprogramban ezenkívül töltsük fel a tömböt számokkal!

```
static void Main()
{
    tomb = new int[N];
    Random rnd = new();
    for(int i=0; i<N; i++)
        tomb[i] = rnd.Next(10000,100000);
}
```

Mivel szeretnénk látni a tömb elemeit rendezés előtt és rendezés után is, ezért érdemes egy `kiir()` nevű metódust létrehozni, ami egymás mellé kiírja a tömb számait. Ha az **N** elemszám 20-30 körüli érték, akkor még elférnek a számok egy sorban, és jól látható a számsorozat. Ezt a metódust a Main() metódus elé helyezzük az osztályban!

```
static void kiir()
{
    for(int i=0; i<N; i++)
        System.Console.Write(tomb[i]+" ");
    System.Console.WriteLine();
}
```

A majd elkészítendő rendezések hatékonysága/sebessége azon múlik, hogy hány alkalommal végeznek összehasonlítást, illetve cserét vagy eltolást a tömb elemin. Ezért minden későbbi rendezés ezt a két számot, mint egy számpárt fogja visszaadni a rendezés eredményeként - természetesen úgy, hogy számolja a rendezés programon belül az összehasonlításokat és a cseréket!

Ennek a számpárnak vegyünk föl egy típust a Rendezések osztály előtt: ez egy struktúra lesz, vagyis egy többelemű, de egy néven elérhető adatszerkezet. Az adatszerkezetben lévő adatmezők nyilvánosak, így azokat más programrészek is elérhetik:

```
struct Par{ public int ossze, csere; }
```

Ha létrehozunk egy `Par` típusú elemet, pl. `Par p = new Par();` paranccsal, akkor a továbbiakban a `p.ossze = 0` paranccsal adhatunk kezdőértéket az összeadások számának, valamint a `p.ossze++;` utasítással növelhetjük az összeadások számát.

Az osztályt a továbbiakban úgy bővítjük, hogy elkészítjük az egyes rendezéseket, majd meghívjuk azokat úgy, hogy visszatérési értékként átvesszük az összeadások és cserék számpárt. Tehát a főprogram a következőkkel bővül. Mivel több rendezést is alkalmaznánk, és szeretnénk mindet ugyanazzal az eredeti adatsorozattal kipróbálni, ezért létrehozunk egy `ment` nevű tömböt, amibe az első rendezés előtt elmentjük a sorozatot.

```
...
int[] ment = (int[]) tomb.Clone(); // elmentjük az eredeti tömböt
Par par = EgyRend(); // egy rendezés, ami rendezi a tömböt
kiir();
System.Console.WriteLine(par.ossze+"--"+par.csere);
...
```

Ezt követően folytathatjuk egy másik rendezéssel, miután az elmentett tömbből visszamásoltuk az eredeti adatsorozatot.

```
...
tomb = (int[]) ment.Clone(); // visszatöltjük az eredeti tömböt
Par par = MasikRend(); // egy rendezés, ami rendezi a tömböt
kiir();
System.Console.WriteLine(par.ossze+"--"+par.csere);
...
```

Az **EgyRend**() és a hasonló rendezések a `Rendezesek` osztály statikus metódusai a `Main`()-hez hasonlóan, azzal a különbséggel, hogy visszaadnak egy `Par` számpárt:

```
Static Par MasikRend()
{
     Par p = new Par();
     p.ossze = 0;
     p.csere = 0;
     return p;
}
```

Az egyes alkalmazott rendezések leírása a többi állományban található. Egyelőre a program az `EgyRend`()-ben ne csináljuk semmit, csak hozza létre a számpárt és adja vissza. Később a megfelelő néven megírt rendezéseket írjuk meg hasonló forma szerint és hívjuk meg őket a főprogramból. A program mostani változatát mentsük el a repo-ba:

git add .

git commit –m “Rendezsek”

git push